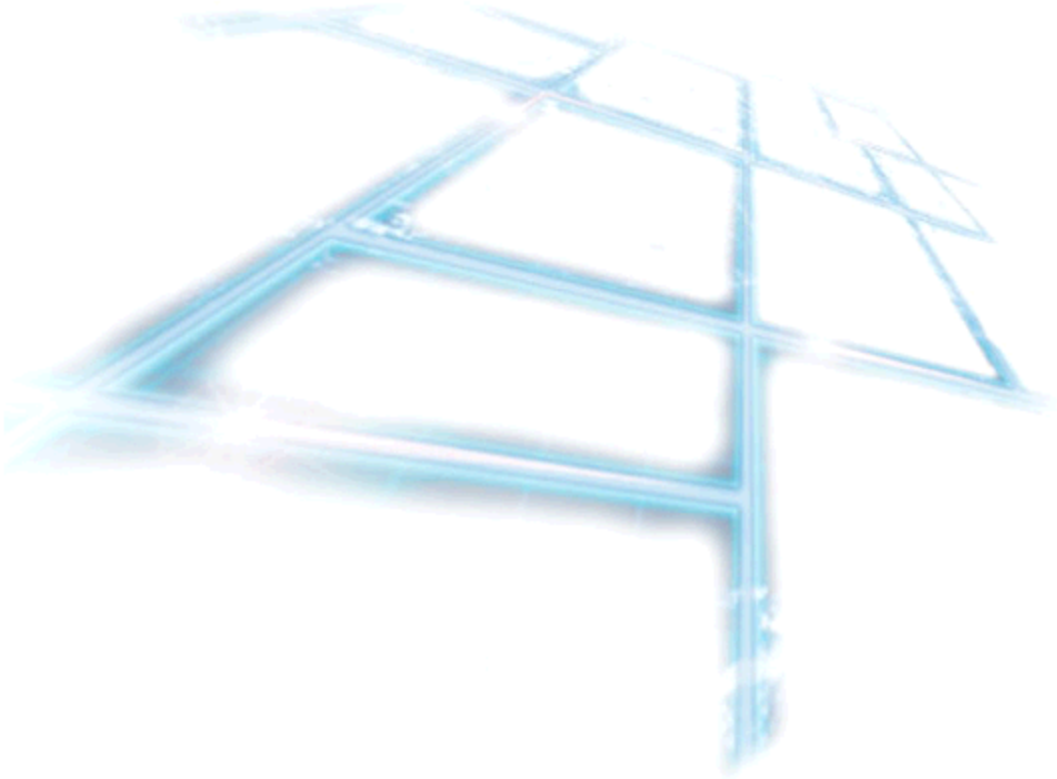
QUANCOM
INFORMATIONSSYSTEME

USBOPTO/REL 64/32/16/8



© QUANCOM Informationssysteme GmbH

Copyright © QUANCOM Informationssysteme GmbH
All specification in this manual was arranged after careful check, and it is not considered as any warranty of product properties. QUANCOM shall not be responsible for any errors or omissions contained in this user's manual, and reserves the right to make changes without notice. Passing on and duplication of this manual and the utilisation of its contents as well as the software belonging to the product are permitted only with written permission by QUANCOM.

Wesseling, April 2009
Version 4.3.1

Index

ChapterI Overview 1

1 Our experience is your profit....1
2 Customer Communication....1
3 Changes in this manual and software updates....1
4 Extend of Delivery....2

ChapterII Installation procedures 3

1 System requirements....3
2 Safety precautions....3
3 Installation of the board....4

ChapterIII Technical hardware description 5

1 Board information....5
General description of the USBOPTO - modules....6
Description of the USBOPTO64IN....6
Description of the USBOPTO64OUT....7
Description of the USBREL64....7
Description of the USBOPTOREL32....7
Description of the USBOPTOREL32/3A....7
Description of the USBOPTO32IO....7
Description of the USBOPTOREL16....8
Description of the USBOPTO16IO....8
Description of the USBOPTOREL8....8
2 Board overview USBOPTO64IN....9
Systembus USB....9
3 Board overview USBREL64 / USBOPTO64OUT....10
Systembus USB....11
4 Board overview USBOPTOREL32 / USBOPTOREL32/3A / USBOPTO32IO....12
Systembus USB....13
5 Board overview USBOPTOREL16 / USBOPTO16IO....14
Systembus USB....15
6 Board overview USBOPTOREL8....16
Systembus USB....16
7 Technical data of the input photocoupler of the modules....18
How to connect the input photocoupler....18
8 Technical data of the output photocoupler(USBOPTO32IO / USBOPTO16IO / USBOPTO64OUT)
How to connect the output photocoupler....19
9 Technical data of the relays (USBREL64 / USBOPTOREL32 / USBOPTOREL32/3A / USBOPTOREL16
How to connect the relays-outputs....20
10 Setting of Power supply (JP1)....21
11 Configuration of the board ID (DIP1)....21
Examples of the choice of the module address....22
12 LED‘s and control elements....22

© QUANCOM Informationssysteme GmbH

ChapterIV Software programming of the modules with the QLIB 23

1 Software....23

Which software do I need?....23

2 QLIB (QUANCOM Driver Library)....24

3 Installation and general programming with the QLIB....25

Installation of the drivers and the QLIB for a QUANCOM PCI board or USB-Module under Windows XP/2000

Installation of the drivers and the QLIB for a QUANCOM PCI board or USB-Module under Windows ME/98 9

ChapterV High-speed installation of the QLIB for USB! 31

ChapterVI QLIB Commands 32

1 Digital write- and read functions....32

Administration- and other functions....33

Digital read functions....36

Digital write functions....37

Administration functions....38

Other functions....39

2 Program examples in Borland Delphi for the QLIB....40

Program examples in C for QLIB....40

Programming the Inputs, Outputs and special functions....41

Programming the photocoupled/relays outputs with the QLIB under C....44

Read the photo-coupled inputs with the QLIB under C....44

Program examples in Borland Delphi for the QLIB....45

Programming the Inputs, Outputs and special functions....46

ChapterVII Annex 51

1 Problems with boards running under Windows 98/95 and Windows XP/2000/NT....51

2 Customer Communication and Help....53

3 Technical support form....55

4 Hardware and software configuration form....56

5 Documentation comment form....57

6 Trademarks....58

© QUANCOM Informationssysteme GmbH

1. Overview

We congratulate you on buying the QUANCOM high quality measurement and automation board. You have chosen a product which attributes and functions show the latest updates of technology.

The following special attributes are included:

- Easy connection through USB
- Easy programming
- Timeout-detection with outputs automatically disabled
- Reliable Input-change-detection
- Various sample applications in different programming languages
- Driver support by Windows XP/2000 and Me/98 with the QLIB **(QUANCOM Driver Library)**

1.1 Our experience is your profit

QUANCOM is specialised in development of hard- and software. QUANCOM has become one of the leading suppliers of measuring and automation technology in industry. At its design centres QUANCOM has developed an impressive range of products.

1.2 Customer Communication

QUANCOM wants to receive your comments on our products and manuals. We are interested in the applications you develop with our products, and we want to help you if you have problems with them. For easy contacting, this manual contains comment and configuration forms for you to complete, which are in chapter ”Customer Communication and Help” at the end of this manual.

1.3 Changes in this manual and software updates

QUANCOM - products are marked out by their constant further development. You can watch all the actual information of the changes in the README-file on the installation disk or CD. You can always get more information and free software updates from our internet website.

© QUANCOM Informationssysteme GmbH

www.quancom.de

1.4 Extend of Delivery

- Measuring and automation module
- User‘s manual
- QUANCOM CD

If a component is missing please contact your dealer. QUANCOM reserves the right to change the extent of delivery without a preliminary announcement.

© QUANCOM Informationssysteme GmbH

2. Installation procedures

2.1 System requirements

- Personal computer: The QUANCOM boards are assigned to operate in IBM-AT compatible computers with 80X86 or compatible. (i.e. Pentium)
- Bus: Your computer must have the corresponding bus. (USB)

More Informations you find in "**Technical hardware description**"

2.2 Safety precautions

For the sake of your security and of a safe function of your new QUANCOM board mind the following advice:

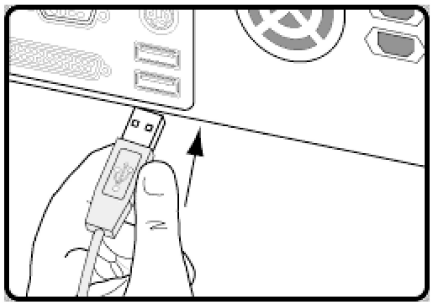
- Before opening the computer please unplug it.
- Computer motherboards and components contain very sensible integrated circuit (IC) chips. You have to obey some precautions whenever you work on your computer to protect them against damage from static electricity. Use a grounded wrist strap before handling computer components. If you don’t have one, touch both of your hands to a safely grounded object or to a metal object, such as the power supply case.
- Take components by the edges and try not to touch the integrated circuit chips, leads or circuitry.
- Place components on a grounded anti-static pad or on the bag that was sent with the component whenever the components are separated from the system.

Modifications of the device made without express permission of QUANCOM, lead to the loss of warranty and certification.

© QUANCOM Informationssysteme GmbH

2.3 Installation of the board

Switch on the PC, start Windows and connect the USB connecting-cable with the USB port of the PC.



© QUANCOM Informationssysteme GmbH

3. Technical hardware description

3.1 Board information

	USBOPTO		USB	USBOPTO		USBOPTO		USB
Photocoupled inputs	**64IN**	**64OUT**	**REL64**	**REL32**	**32IO**	**REL16**	**16IO**	**REL8**
Relays-outputs	64	-	-	32	32	16	16	8
Photocoupled outputs	-	-	64	32	-	16	-	8
LED visual state control	-	64	-	-	32	-	16	-
	64	64	64	64	64	32	32	16

- Resistornetwork adjustable input voltage range.
- Timeout detection (Watchdog) which automatically disables the outputs
- Quick and precise acquisition of switched events
- Simple programming of the module with the QLIB
- Connection via USB
- Power supply via USB or external
- Use of a maximum of 16 Modules
- Simple packaging i.e. on a mounting rail, as a stand alone device or as mobile set for the laptop computers usable.

© QUANCOM Informationssysteme GmbH

3.1.1 General description of the USBOPTO - modules

Input-protection for up to 500V through photocouplers

The QUANCOM® USBOPTO-modules have photocoupled inputs and relays or photocoupled outputs. With the use of photocoupled in- and outputs (protection of the PC from overvoltage or wrong polarity) the safety was increased through an internal watchdog.

Detecting changes of input signals

The modules are equipped with an integrated detection logic which detects fast input changes surely. The input voltages are adjustable through an plug able resistor network, so you may chose if you want to use an input range from 12V to 30V or from 5V to 12V. You may tell us your needs with your order and we pre-set up your module.

Separated input and output photocouplers

Through consequent separating input and output signals you don’t need to obey any common potential. Each of the input and output channels has got its own two-wire connection to the connector. You see you needn’t even to watch out for shortcuts or other wrong potentials.

LED’s as visual status for each input and output signal

You can read all input and output signal states through LED’s located at the module. Each access to the module is also displayed with an LED and you can see its configuration state.

Comfortable Programming

Because we implemented support for these modules into the QLIB (QUANCOM driver library) you do not need to write your own drivers for these modules. You can access the module through the QLIB simply with an easy-to-use API.

3.1.1.1 Description of the USBOPTO64IN

The USBOPTO64IN is a USB modul with 64 photocoupled inputs.

© QUANCOM Informationssysteme GmbH

3.1.1.2 Description of the USBOPTO64OUT

The USBOPTO64OUT is a USB modul with 64 photocoupled outputs.

3.1.1.3 Description of the USBREL64

The USBREL64 is a USB modul with 64 relays.

3.1.1.4 Description of the USBOPTOREL32

The USBOPTOREL32 is a USB modul with 32 photocoupled inputs und 32 SIL-Reed Relays. You are able to rate the relay with 1 A or 24 W .

3.1.1.5 Description of the USBOPTOREL32/3A

The USBOPTOREL32 is a USB modul with 32 photocoupled inputs und 32 SIL-Reed Relays. You are able to rate the relay with 3 A or 90 W .

3.1.1.6 Description of the USBOPTO32IO

The USBOPTOREL32 is a USB modul with 32 photocoupled inputs und 32 photocoupled outputs. You are able to rate the outputs with 100 mA .

© QUANCOM Informationssysteme GmbH

3.1.1.7 Description of the USBOPTOREL16

The USBOPTOREL32 is a USB modul with 16 photocoupled inputs und 16 SIL-Reed Relays. You are able to rate the relay with 1 A or 24 W .

3.1.1.8 Description of the USBOPTO16IO

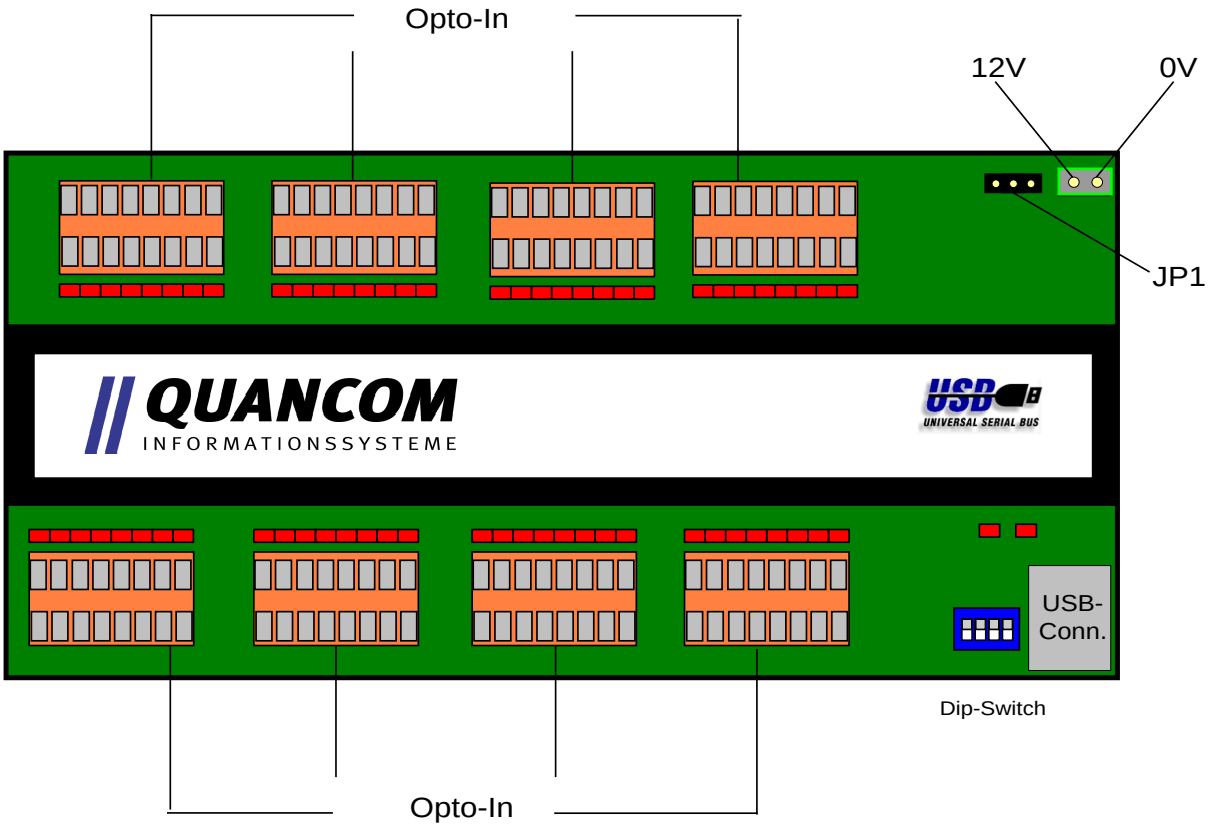
The USBOPTOREL32 is a USB modul with 16 photocoupled inputs und 16 photocoupled outputs. You are able to rate the outputs with 100 mA .

3.1.1.9 Description of the USBOPTOREL8

The USBOPTOREL32 is a USB modul with 8 photocoupled inputs und 8 SIL-Reed Relays. You are able to rate the relay with 1 A or 24 W .

© QUANCOM Informationssysteme GmbH

3.2 Board overview USBOPTO64IN

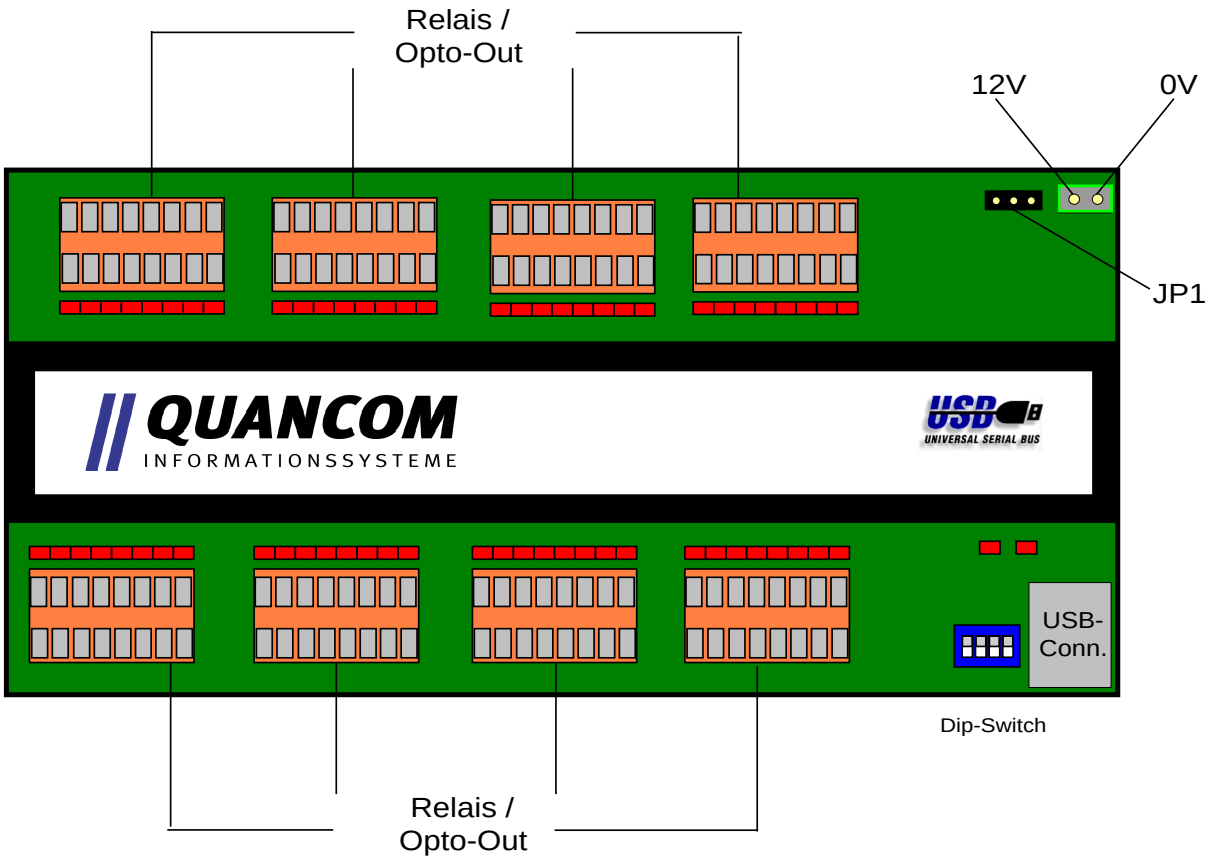


3.2.1 Systembus USB

	USBOPTO64IN
Inputs	64 photo isulated (15 V – 30 V) (5 V - 15 V optional)
Input voltage range	-
Input current range	-
Opto-Out	-

© QUANCOM Informationssysteme GmbH

3.3 Board overview USBREL64 / USBOPTO64OUT



3.3.1 Systembus USB

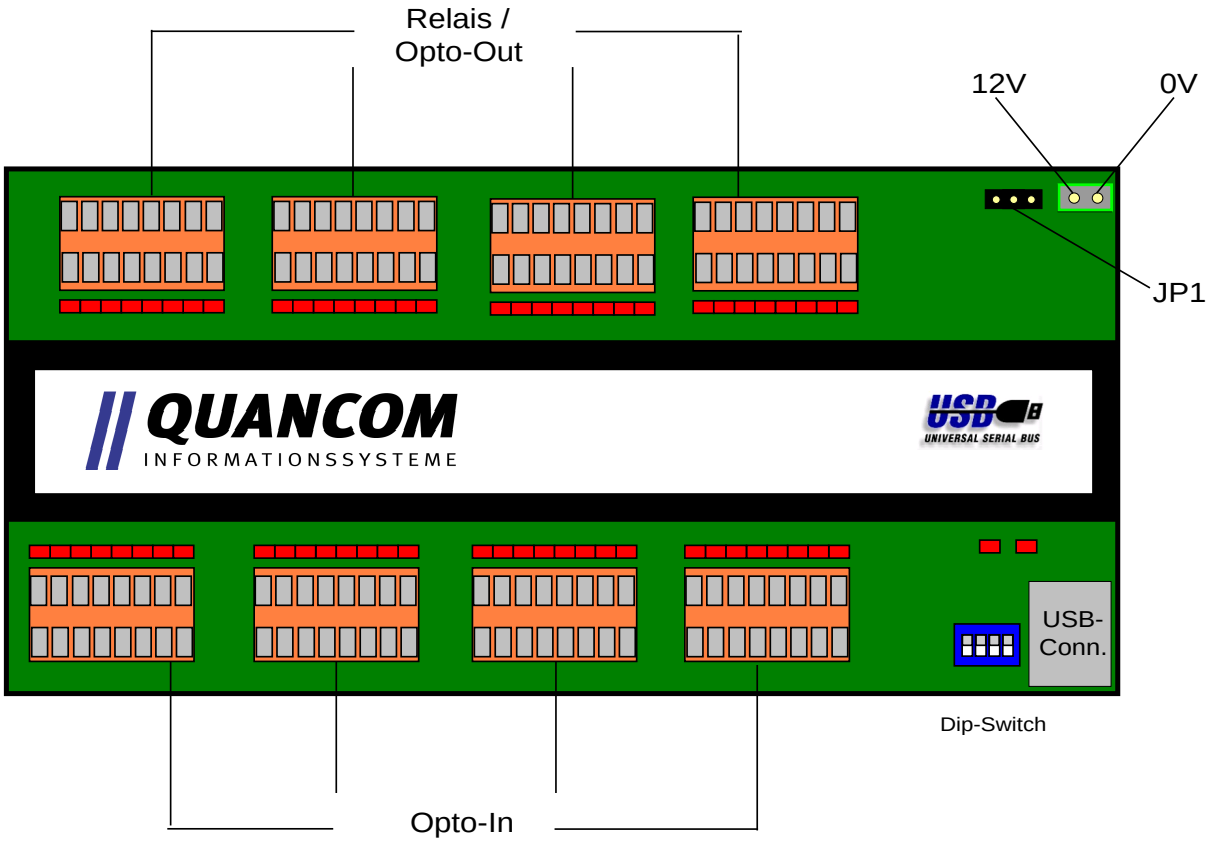
	USBREL64	OPTO64OUT
Input voltage range	Up to 40V	0...30 V
Input current range	1A	max. 100mA per chanel
Relays	32 switch-on relays maximal 1A / 24W	-
Switching time	1 ms	-
Opto-Out	-	64 photo isolated (24V)

- Systembus USB
- CE Yes
- Timeout-detction Yes (0,35s – 44,7s in Steps)
- Temperature 0...70°C

The USBREL64 and the USBOPTO64OUT should be always use with an external power supply.

© QUANCOM Informationssysteme GmbH

3.4 Board overview USBOPTOREL32 / USBOPTOREL32/3A / USBOPTO32IO



3.4.1 Systembus USB

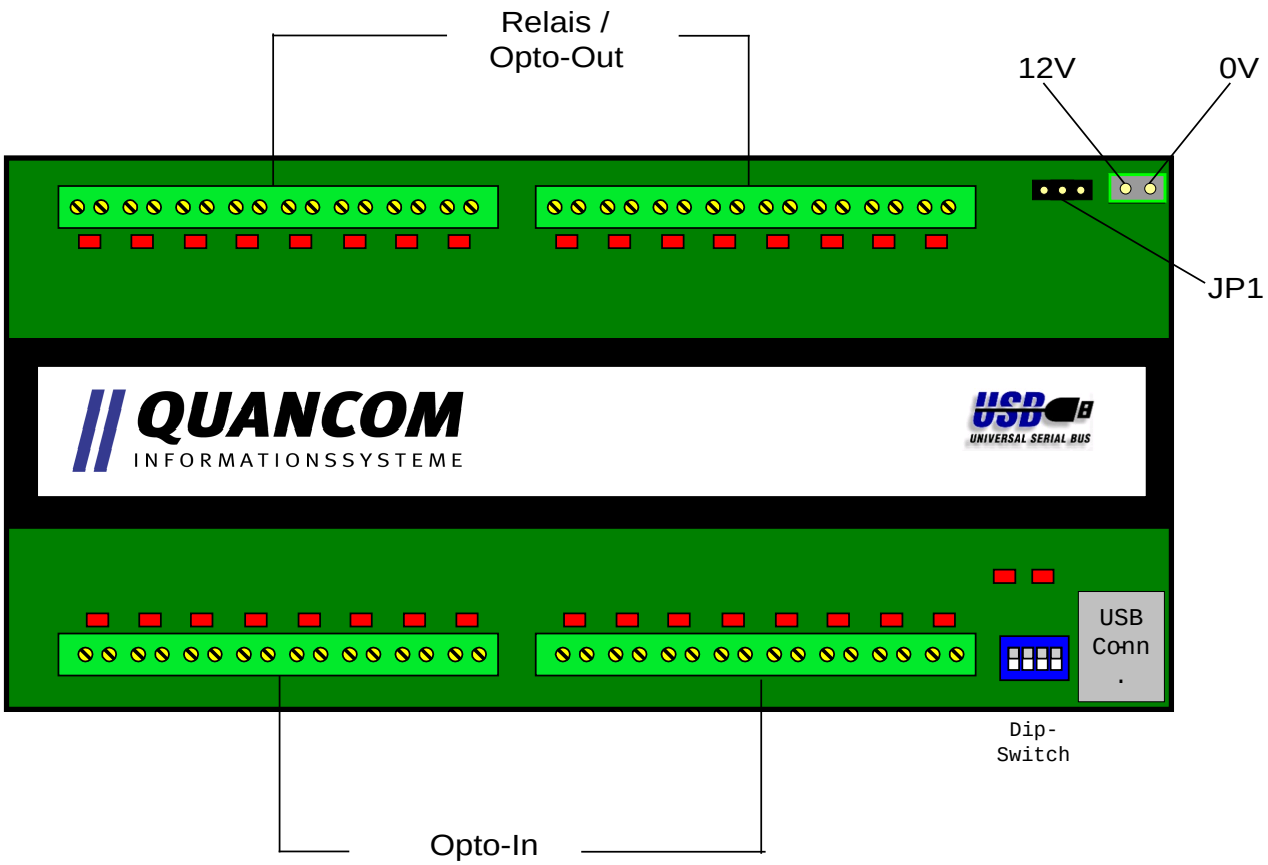
	USBOPTOREL32	USBOPTOREL32/3A	USBOPTO32IO
Inputs	32 Photo isolated (15 V – 30 V) (Optional 5 V - 15 V)	32 Photo isolated (15 V – 30 V) (Optional 5 V - 15 V)	32 Photo isolated (15 V – 30 V) (Optional 5 V - 15 V)
Input voltage range	Up to 40V	Up to 40V	0...30 V
Input current range	1A	3A	max. 100mA per Channel
Relays	32 Relays Maximal 1A / 24W	32 Relays Maximal 3A / 90W	-
Switching Time	1 ms	1ms	-
Photo-Out	-	-	32 photo isolated (24V)

- Systembus USB
- Timeout-detection Yes (0,35s – 44,7s in Steps)
- CE Yes
- Temperature 0...70°C

The USBREL64 and the USBOPTO64OUT should be always used with an external power supply.

© QUANCOM Informationssysteme GmbH

3.5 Board overview USBOPTOREL16 / USBOPTO16IO



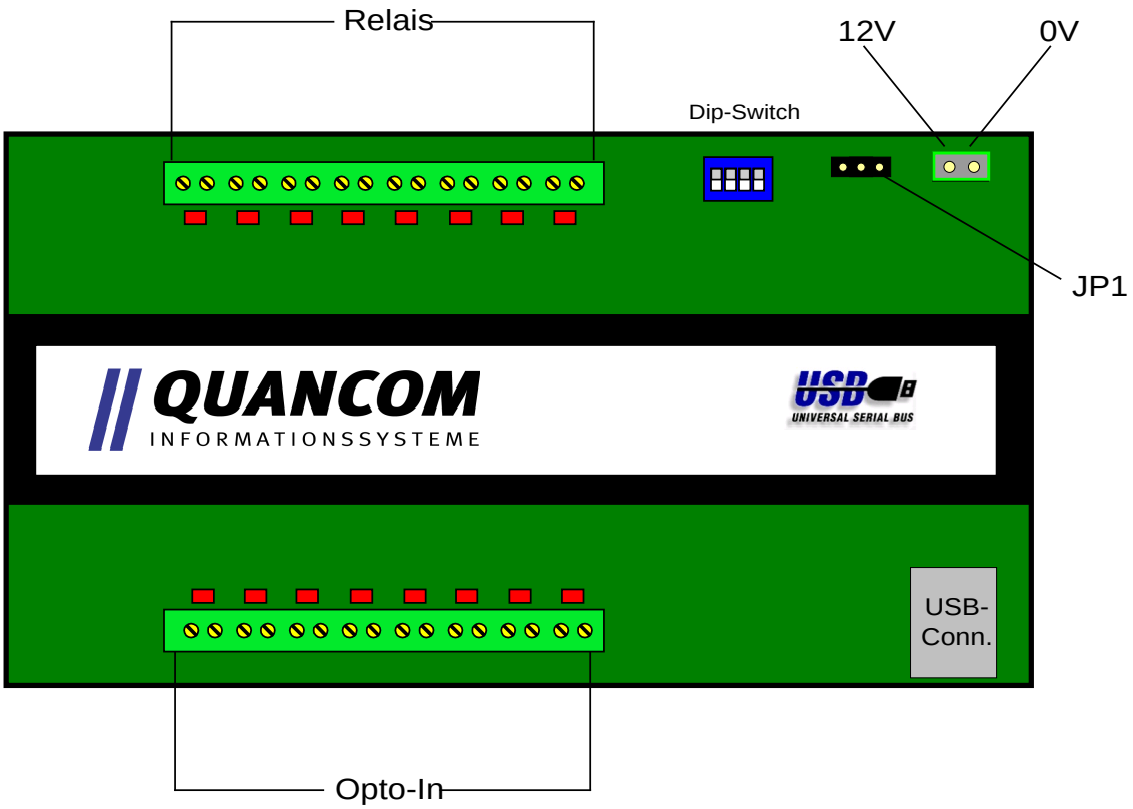
3.5.1 Systembus USB

	USBOPTOREL16	USBOPTOREL16IO
Inputs	16 photo isulated (15 V – 30 V) (5 V - 15 V optional)	16 photo isulated (15 V – 30 V) (5 V - 15 V optional)
Input voltage range	Up to 40V	0...30 V
Input current range	1A	max. 100mA per chanel
Relays	32 switch-on relays maximal 1A / 24W	-
Switching time	1 ms	-
Opto-Out	-	32 photo isulated (24V)

- Systembus USB
- Timeout-detection Yes (0,35s – 44,7s in Steps)
- CE Yes
- Temperature 0...70°C

© QUANCOM Informationssysteme GmbH

3.6 Board overview USBOPTOREL8



3.6.1 Systembus USB

	USBOPTOREL8
Inputs	8 photo isulated (15 V – 30 V) (5 V - 15 V optional)
Input voltage range	Up to 40V
Input current range	1A
Relays	8 switch-on relays maximal 1A / 24W
Switching time	1 ms
Opto-Out	-

© QUANCOM Informationssysteme GmbH

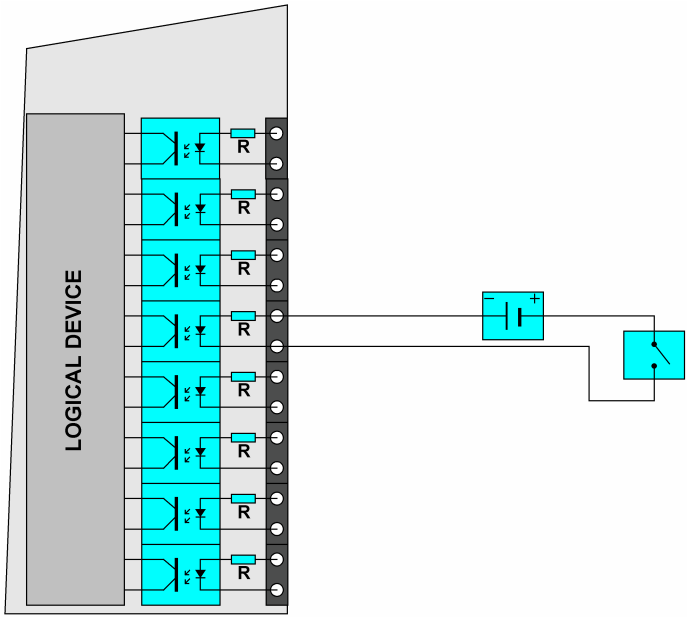
- Systembus USB
- Timeout-detection Yes (0,35s – 44,7s in Steps)
- CE Yes
- Temperature 0...70°C

© QUANCOM Informationssysteme GmbH

3.7 Technical data of the input photocoupler of the modules

Maximum input voltage	5-15V, 15-30V
Maximum current	10mA

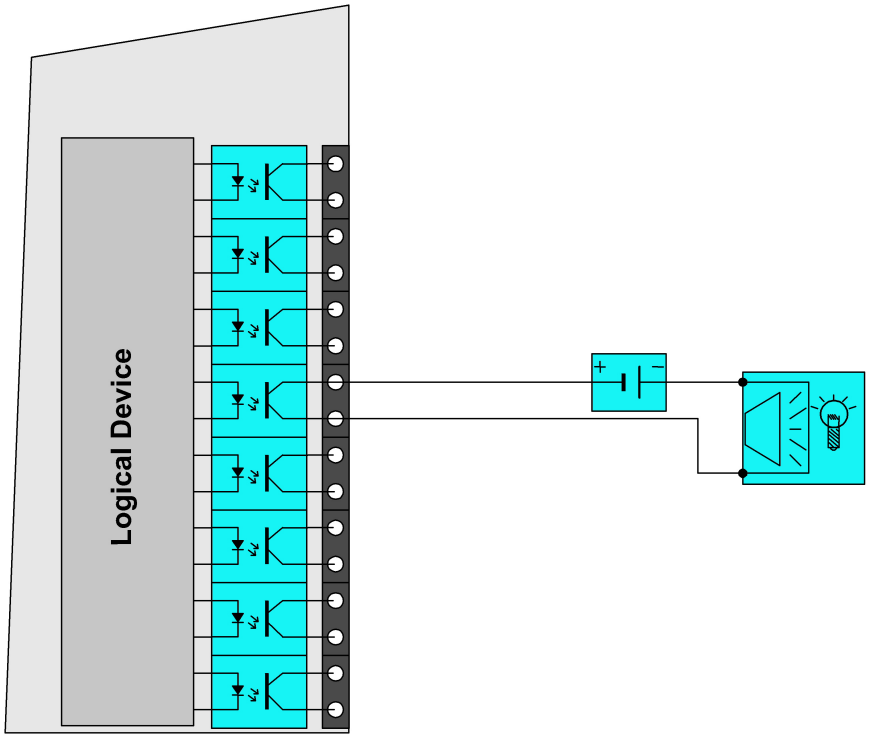
3.7.1 How to connect the input photocoupler



3.8 Technical data of the output photocoupler(USBOPTO32IO / USBOPTO16IO / USBOPTO64OUT)

Maximum current	100 mA per Channel
Maximum voltage	0-30V DC

3.8.1 How to connect the output photocoupler

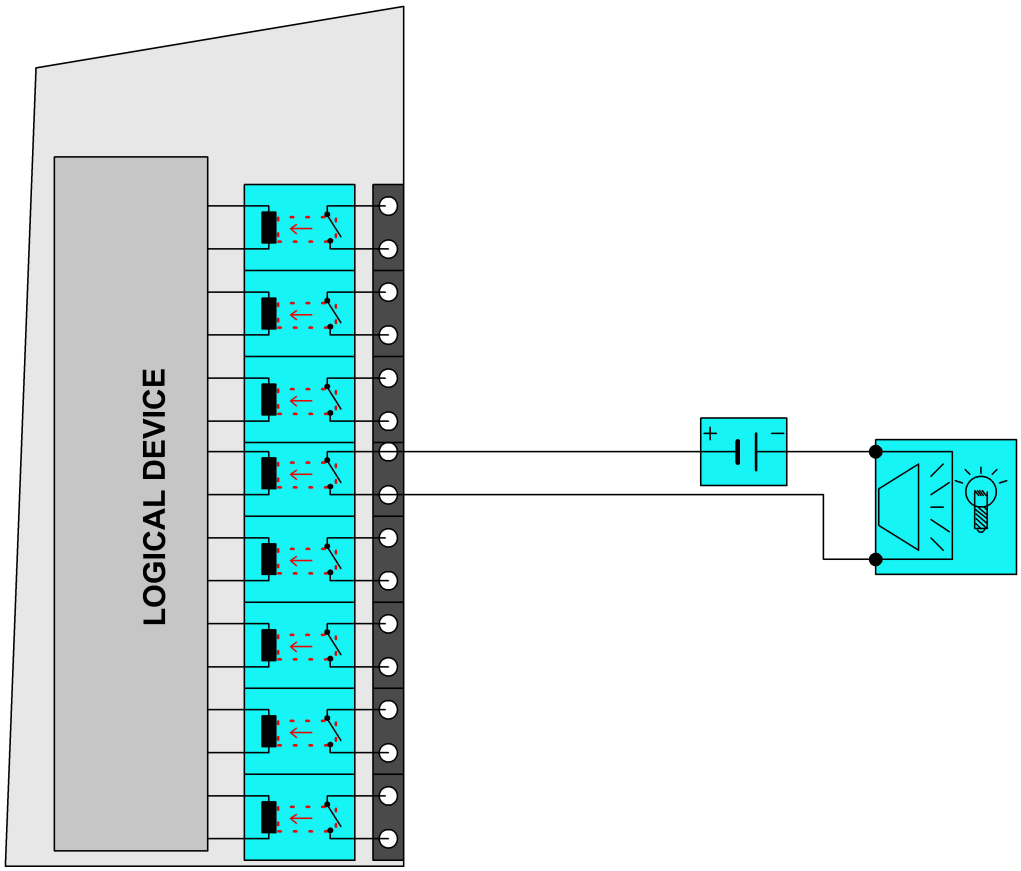


© QUANCOM Informationssysteme GmbH

3.9 Technical data of the relays (USBREL64 / USBOPTOREL32 / USBOPTOREL32/3A / USBOPTOREL16)

Maximum current / power (USBREL64)	1 Ampere / 15 Watt
Maximum current / power (USBOPTOREL32)	1 Ampere / 24 Watt
Maximum current / power (USBoptorel32/3A)	3 Ampere / 90 Watt 1 Ampere / 24 Watt
Maximum current / power (USBOPTOREL16)	1 Ampere / 15 Watt
Maximum switching voltage	40V DC
Switching time (with chatter)	1ms

3.9.1 How to connect the relays-outputs



© QUANCOM Informationssysteme GmbH

3.10 Setting of Power supply (JP1)

You can chose the way of the power supply between USB or external 12 V connector by setting of Jumper 1 (JP1).

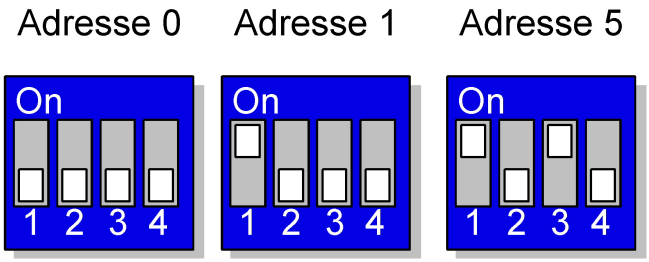
3.11 Configuration of the board ID (DIP1)

You can use a maximum of 16 USB-moduls with one Computer.
If you use more then one of the same USB Board, please set a different ID for each module with the DIP-Switch DIP1

SW1	SW2	SW3	SW4	Card ID
Off	Off	Off	Off	0
On	Off	Off	Off	1
Off	On	Off	Off	2
On	On	Off	Off	3
Off	Off	On	Off	4
On	Off	On	Off	5
Off	On	On	Off	6
On	On	On	Off	7
Off	Off	Off	On	8
On	Off	Off	On	9
Off	On	Off	On	10
On	On	Off	On	11
Off	Off	On	On	12
On	Off	On	On	13
Off	On	On	On	14
On	On	On	On	15

© QUANCOM Informationssysteme GmbH

3.11.1 Examples of the choice of the module address



3.12 LED‘s and control elements

You can inspect the **actual state** of every input and output with the LED´s.

There are additional LED´s on the board for :

- **Read and Write Access**

- Indicates (LED is on) a read or write access to the board. (i.e. an access to the relays, photocoupler or the control-register)

- **Input Change**

- Indicates, if (LED is on) one or more inputs have changed their state. The LED dins, if the change is acknowledged by the PC.

- **Time Out**

- Indicates that (LED is on) a time out event happens. The LED dins, if the change is acknowledged by the PC.

- **Power +5V**

- Indicates that (LED is on) the module is connected to the power supply.

- **Activated**

- Indicates (LED is on) that the PCI-device driver enumerates the board.

© QUANCOM Informationssysteme GmbH

4. Software programming of the modules with the QLIB

4.1 Software

4.1.1 Which software do I need?

The software which I need is depends on respective application and the respective operating system. To obtain by access by program to the map the following possibilities exist:

Method 1:
High level programming (access to the map over the QLIB) under Windows XP/2000/NT4.x/ME/98/95. Hereby it is possible to address the map e.g. over Visual C, Visual basic, boron country Delphi, Lotus Notes among other things for compilers and interpreters.

Method 2:
Installation of the QLIB in connection with another program. Examples for the employment with Labview of national instrument will be find after the installation of the QLIB in the listing of samples \ usbopto the QLIB listing.

If you liked to use method 1, you need the source text of application. They are even responsible for adding the instructions into your application. Around these methods to use programming knowledge is necessary.

Method 2 permits to make work the QUANCOM map with an existing software e.g. LabView or Agilent VEE. But you must install the QLIB of the installation CD as first. References approximately around the QLIB and installation please take from the QLIB manual, which likewise on that CD. On that CD find you at the same time some sample programs for LabView and Agilent VEE.

© QUANCOM Informationssysteme GmbH

4.2 QLIB (QUANCOM Driver Library)

The **QLIB**, which stands for **QUANCOM Driver LIBRARY**, was developed with the target to allow the simple programming of all our data acquisition products under various operating systems. So it is easy to write an application that runs under the operating systems Windows Me/98/95 and Windows XP/2000/NT4.0. This driver interface is not limited to PC boards or other I/O adapters but is also targeted towards supporting the next product generations currently being developed. The used functions and parameters are the same for all operating systems.

Supported operating systems:

- <%OPERATINGSYS_ALL%>

Compiler:

C / C++

- Borland C++ 3.1, 4.x, 5.x
- Microsoft® Visual C++ 1.x, 2.x, 4.x, 5.x, 6.x

Pascal

- Borland Turbo Pascal

Delphi

- Borland Delphi

Basic

- Microsoft® Visual Basic 3.x, 4.x, 5.x; 6.x

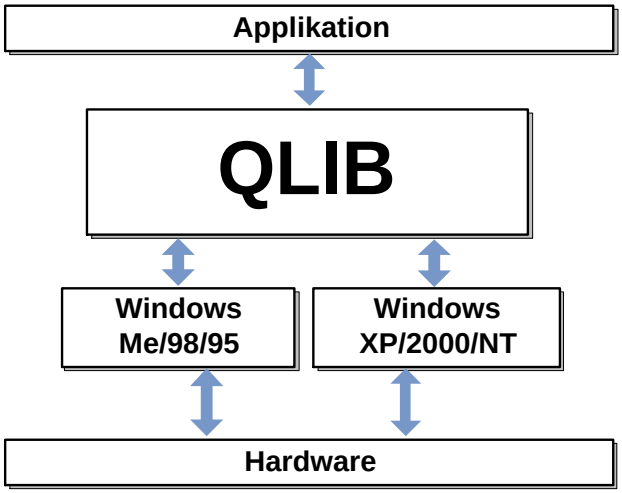
Graphical Programming Language

- Agilent VEE from Agilent Technologies
- LabView® from National Instruments

© QUANCOM Informationssysteme GmbH

4.3 Installation and general programming with the QLIB

For further information about the installation process and the general programming with the QLIB (i.e. setting up the system, linking libraries, adding header files, etc.) please read the “ QLIB” manual or the windows help file (qlib.hlp) which is included on the installation CD.
The following chapters describe the installation of the drivers depending on the operating system (Windows Me/98 or Windows XP/2000). The installation procedure differs for every operating system.



© QUANCOM Informationssysteme GmbH

Please program all QUANCOM-boards independently. The QLIB (QUANCOM LIBRARY) offers the possibility of addressing all QUANCOM-boards under the operating systems Windows XP/2000/NT and ME/98/95 and the programming languages C/C++/Delphi/Visual Basic with simple instructions. The QLIB is shipped with all QUANCOM-Boards and simplifies the integration into own applications for the user.

There are special Installation instructions for different operating systems and bussystems (PCI, ISA, USB, PAR). Please choose the proper Installation for your board and your operating system at the following chapters.

4.3.1 Installation of the drivers and the QLIB for a QUANCOM PCI board or USB-Module under Windows XP/2000

If you downloaded the QLIB from our internet page, please read the information inside the Readme-file from your downloaded file before you install the drivers.

The software-install process depends on administrator-rights in order to install services and driver-files. If you don't have such rights, contact you local system-administrator.

1. Step : Driver installation:

After installing the board into a free Slot, please restart your system. Windows will recogni the new board automatically and will demand a driver for the new board. Please insert the **QLIB Installation CD** in your CD Drive.

- If the dialog box "new hardware found" appears, choose "Next".
- After that, you have to select "search for a suitable driver for the equipment (recommended)", and confirm it with "Next".
- As source for the driver's search you have to select "Choose the destination" and press "Next".
- Windows opens a dialogue box to choose an file now. Please click on the button "Choose"

© QUANCOM Informationssysteme GmbH

, change to the CD-Drive and choose the folder **Win2000** or **WinXP**, depending on your operating system. Windows will find the file **QLIBXDRV.INF** automatically. Please click on “ OK” to go ahead with the Installation of the driver.

- Now, you have to confirm with ”next” that you want to install the driver.
- In the following window you finish the installation of the QUANCOM driver with click on the button ”finish”.

After the installation of the QUANCOM board driver you have to install the **QLIB Software**.

1. Step : QLIB installation:

- Please click on **Start | Run** and choose the program qlib32.exe from the installation-CD, click on “OK”.
- If a message box appears after starting the program Qlib32.exe, which states that your system must be restarted to install the QLIB, please click on “YES” to restart your system. After the reboot, the installation will be automatically continued

(This step is only necessary if you had no installer version on your system)

- You have to click on the “Next” button in the following window to continue the installation.

(Please consider that all current windows programs are terminated before starting the installation)

- Then, please confirm the license agreement of the QLIB-Software with a click on “I accept the license agreement” and lead the installation with pressing the “Next” button.
- Please enter your personal user data (name; organisation;) now. Select whether you want to install the QLIB-Software only for the current user or for all users of this computer (only possible with administrator rights) afterwards and confirm your choice with a click on “Next” .
- Please click on “Browse” to change the installation path of the QLIB-Software The installation will be continued with clicking on “Next”.
- Please select the kind of software installation now and confirm it with “Next”. Three possibilities are available :

- **Typical** (without examples and help files)
- **Complete** (with examples and help files)

© QUANCOM Informationssysteme GmbH

- **Custom** (free selectable)

- The installation starts immediately with a click on “Next”.
- After the successful installation of the QLIB-Software, the installation program will be closed with a click on the “Finish” button.
- You will be requested to restart your system to overtake the changes at your computer. This happens when you press the “YES” Button.

© QUANCOM Informationssysteme GmbH

4.3.2 Installation of the drivers and the QLIB for a QUANCOM PCI board or USB-Module under Windows ME/98

If you downloaded the QLIB from our internet page, please read the information inside the Readme-file from your downloaded file before you install the drivers.

1. Step : Driver installation:

After putting the board into a free slot, please restart your system. Windows will recognise the new board automatically and will demand a driver for the new founded board. Please put the **QLIB Installation CD** in your CD Drive.

- If Windows indicates the dialogue box "new hardware found" when starting, then select "Next".
- Then you have to select "search for a suitable driver for the equipment (recommended)" and then confirm your decision with "Next".
- As source for the driver's search you have to select "Choose the destination" and press "Next".
- Windows opens a dialogue box to choose a file now. Please click on the button "Choose", change to the CD-Drive and choose the folder **WinME**, or **Win98**, it depends on the Operating Systems in which you want to install the QLIB. Windows will find automatically the file QLIBXDRV.INF. Please click on "OK" to go on with the driver's installation.
- You have to confirm that you want to install the driver by clicking the "Next"- button.
- In the following window you close the QUANCOM driver installation with a click on the button "Finish".

1. Step : QLIB installation:

After the driver is installed on the QUANCOM board you have to install the **QLIB Software.**

- Please click on **Start | Execute** and choose the program qlib32.exe from the installation-CD, click on "OK"
- If a message box appears after starting the program Qlib32.exe, which states that your system must be restarted to install the QLIB, please click on "YES" to restart your system. After the reboot, the installation will be automatically continued.

(This step is only necessary if you had no installer version on your system)

© QUANCOM Informationssysteme GmbH

- You have to click on the “Next” button in the following window to continue the installation. (Please consider that all current windows programs are terminated before starting the installation)
- Then, please confirm the license agreement of the QLIB-Software with a click on “I accept the license agreement” and lead the installation with pressing the “Next” button.
- Please enter your personal user data (name; organisation;) now. Select whether you want to install the QLIB-Software only for the current user or for all users of this computer afterwards and confirm your choice with a click on “Next”.
- Please click on “Browse” to change the installation path of the QLIB-Software The installation will be continued with clicking on “Next”.
- Please select the kind of software installation now and confirm it with “Next”. Three possibilities are available :

- **Typical** (without examples and help files)
- **Complete** (with examples and help files)
- **Custom** (free selectable)

- The installation starts immediately with a click on “Next”.
- After the successful installation of the QLIB-Software, the installation program will be closed with a click on the “Finish” button.
- You will be requested to restart your system to overtake the changes at your computer. This happens when you press the “YES” button

© QUANCOM Informationssysteme GmbH

5. High-speed installation of the QLIB for USB!

For the installation of the drivers and run time environment administrator rights are necessary. Without the appropriate rights the driver and the run time environment cannot be installed correctly.

Windows XP/2000	Windows ME/98
USB module input	USB module put in 2
start Computers	start Computers
USB version recognized by the operating system, please the path to the drivers to indicate. (Drivers are in the listing WinXP or Win2000 that installation CD)	USB version recognized by the operating system, please the path to the drivers to indicate. (Drivers are in the listing WinME or Win98 that installation CD)
QLIB, test routines and examples install Start (in the general statement that installation CD QUANCOM.EXE)	QLIB, test routines and examples install Start (in the general statement that installation CD QUANCOM.EXE)
reboot Computers	reboot Computers

© QUANCOM Informationssysteme GmbH

6. QLIB Commands

Make sure that the QLIB (QUANCOM Driver Library) is properly installed. For further information about the installation and how to include the necessary files in your application see the “QLIB” documentation. This chapter describes the special commands that are required to use a QUANCOM board with the QLIB (QUANCOM Driver Library).

6.1 Digital write- and read functions

QAPIPutDO

ULONG **QAPIPutDO** (ULONG **cardid** ULONG **channel** ULONG **value**);

It is possible to give out a 32 Bit width digital value on a channel of the DO card with the function QAPIExtWriteDO32

Parameter

cardid

channel

QAPIGetDI

ULONG **QAPIGetDI** (ULONG **cardid** ULONG **channel**);

The condition of a 32 Bit width digital channel could be read by a DI card with the function QAPIGetDI

Parameter

cardid

© QUANCOM Informationssysteme GmbH

channel

6.1.1 Administration- and other functions

QAPINumOfCards

ULONG **QAPINumOfCards** (void);

It is possible to ask , which used cards are supported by the QLIB with the function QAPINumOfCards.

QAPIGetCardInfo

LPCARDDATAS **QAPIGetCardInfo** (ULONG **cardid**);

It is possible to get some information about the card with the function QAPIGetCardInfo.

QAPIGetCardInfoEx

ULONG **QAPIGetCardInfoEx** (ULONG **cardid** LPCARDDATAS **lpcd**);

It is possible to get some information about the card memory with the function QAPIGetCardInfoEx. These will be written into the applications.

QAPISpecial

ULONG QAPISpecial (ULONG cardid, ULONG jobcode, ULONG para1, ULONG para2);

It is possible to execute special functions of the board with the function QAPISpecial.

© QUANCOM Informationssysteme GmbH

The following parameter for the **USBOPTOREL32 / USBOPTO32IO / USBOPTOREL16 / USBOPTO16IO** of the QAPISpecial function exists:

jobcode	Function	*para1*	*para2*	Return value
JOB_READ_IN_FFS USBPTO64IN/OUT / USBOPTOREL32 (3A) / USBOPTO32IO / USBOPTOREL16 / USBOPTO16IO/ USBOPTOREL8	This function reads the state of all 8/ 16 / 32 / 64 input flipflops (the bits 0 –7/15/31)are set, if the input has changed between two read instructions), bit 0 belongs to channel IN1 / 32 and bit 15/31 belongs to channel IN16 / 32 /64	---	0/1	Input – FF 8/16/32 bit# Para2 is for the 64 moduls 0 means the bit 0-31 and if para2 is 1 it means the bit 32-63
JOB_READ_IN_FFS (Only for USBOPTOREL32 / USBOPTO32IO)	This function reads the state of all 32 input flipflops (the bits 0 – 31 are set, if the input has changed between two read instructions), bit 0 belongs to channel IN1 and bit 31 belongs to channel IN32	---	---	Input – FF (32 bits)
JOB_ENABLE_TIMEOUT	Enables the timeout function. The timeout function disables all outputs, after a certain time, where no access took place.	---	---	---
JOB_DISABLE_TIMEOUT	Disables the timeout function	---	---	---
JOB_RESET_TIMEOUT_STATUS	Resets the timeout status flag. The timeout status flag is set if the timeout state has been entered. In the timeout state all outputs are disabled.	---	---	---
JOB_READ_TIMEOUT_STATUS	Reads the timeout status flag. The timeout status flag is set if the timeout state has been entered. In the timeout state all outputs are disabled.	---	---	1 Timeout state entered 0 Normal operation

© QUANCOM Informationssysteme GmbH

JOB_SET_WATCHDOG_TIME	Sets the timeout interval. The timeout function disables all outputs, after a timeout interval where no access took place.	0 = 0.35s 1 = 0.7s 2 = 1.4s 3 = 2.8s 4 = 5.6s 5 = 11.2s 6 = 22.4s 7 = 44.7s	---	---
JOB_READ_DIP _SWITCH	Shows the settings of the DIP switch of the USB board.	---	---	0...15 DIP Switch Settings

© QUANCOM Informationssysteme GmbH

6.1.2 Digital read functions

QAPIExtReadDI1

ULONG **QAPIExtReadDI1** (ULONG **cardhandle** ULONG **channel** ULONG **mode**);

The condition of a 1 Bit width digital channel could be read by a DI card with the function QAPIExtReadDI1.

QAPIExtReadDI8

ULONG **QAPIExtReadDI8** (ULONG **cardhandle** ULONG **channel** ULONG **mode**);

The condition of a 8 Bit width digital channel could be read by a DI card with the function QAPIExtReadDI8.

QAPIExtReadDI16

ULONG **QAPIExtReadDI16** (ULONG **cardhandle** ULONG **channel** ULONG **mode**);

The condition of a 16 Bit width digital channel could be read by a DI card with the function QAPIExtReadDI16

QAPIExtReadDI32

ULONG **QAPIExtReadDI32** (ULONG **cardhandle** ULONG **channel** ULONG **mode**);

The condition of a 32 Bit width digital channel could be read by a DI card with the function QAPIExtReadDI32

© QUANCOM Informationssysteme GmbH

6.1.3 Digital write functions

QAPIExtWriteDO1

void **QAPIExtWriteDO1** (ULONG **cardhandle** ULONG **channel** ULONG **value** ULONG **mode**);

It is possible to give out a 1 Bit width digital value on a channel of the DO card with the function QAPIExtWriteDO1

QAPIExtWriteDO8

void **QAPIExtWriteDO8** (ULONG **cardhandle** ULONG **channel** ULONG **value** ULONG **mode**);

It is possible to give out a 8 Bit width digital value on a channel of the DO card with the function QAPIExtWriteDO8.

QAPIExtWriteDO32

void **QAPIExtWriteDO16** (ULONG **cardhandle** ULONG **channel** ULONG **value** ULONG **mode**);

It is possible to give out a 16 Bit width digital value on a channel of the DO card with the function QAPIExtWriteDO16.

QAPIExtWriteDO32

© QUANCOM Informationssysteme GmbH

void **QAPIExtWriteDO32** (ULONG **cardhandle** ULONG **channel** ULONG **value** ULONG **mode**);

It is possible to give out a 32 Bit width digital value on a channel of the DO card with the function QAPIExtWriteDO32.

6.1.4 Administration functions

QAPIExtOpenCard

ULONG **QAPIExtOpenCard** (ULONG **cardid**, ULONG **devnum**);

Use the function QAPIExtOpenCard to open a board and retrieve the board handle

QAPIExtCloseCard

void **QAPIExtCloseCard** (ULONG **cardhandle**);

The function QAPIExtCloseCard will close the board.

QAPIExtNumOfCards

ULONG **QAPIExtNumOfCards** (void);

It is possible to ask , which used cards are supported by the QLIB with the function QAPIExtNumOfCards

QAPIExtSpecial

ULONG **QAPIExtSpecial** (ULONG **cardhandle**, ULONG **jobcode**, ULONG **para1**, ULONG

© QUANCOM Informationssysteme GmbH

para2);

It is possible to execute special functions of the board with the function QAPIExtSpecial.

6.1.5 Other functions

QAPIExtGetCardInfo

LPCARDDATAS **QAPIExtGetCardInfo**(ULONG **cardid**);

It is possible to get some information about the card with the function QAPIExtGetCardInfo.

QAPIExtGetCardInfoEx

ULONG **QAPIExtGetCardInfoEx**(ULONG **cardid** LPCARDDATAS **lpcd**);

It is possible to get some information about the card with the function QAPIExtGetCardInfoEx. These will be written into the applications memory

QAPIExtReleaseCardInfo

void **QAPIExtReleaseCardInfo**(LPCARDDATAS **carddatas**);

It is possible with QAPIExtGetCardInfo to get out the asked card information with the function QAPIExtReleaseCardInfo

© QUANCOM Informationssysteme GmbH

6.2 Program examples in Borland Delphi for the QLIB

After the installation of the QLIB, you'll find the example source (usbmodul.pas) at the folder d:\program files\quancom\qlib32\samples\usbopto\delphi (only if you had installed to the standard folder).

6.2.1 Program examples in C for QLIB

The following examples show how easy it is to program with the QLIB.

© QUANCOM Informationssysteme GmbH

6.2.1.1 Programming the Inputs, Outputs and special functions

After installing the QLIB, you will find the example source at the folder d:\program files\quancom\qlib32\samples\usbopto\vc (only if you had installed it into the standard folder).

Please check the installing information's at the header of the source code.

```
// usbmodul.cpp : Sample project for the USBOPTOREL16 and USBOPTOIO Modules
//
// Author: Michael Reimer, QUANCOM Informationssysteme GmbH, Germany
//
// Website: http://www.quancom.de
// Product:
// USB OPTO I/O Module http://www.quancom.de/qprod01/eng/pb/usbopto16io.htm
// USB OPTO Relay Module http://www.quancom.de/qprod01/deu/pb/usboptorel16.htm
// Information:
//
// To use the QLIB Commands in your source, do the following:
//
// (1) Add statement #include "qlib.h" to your source file.
// (2) Add in the Dialog Menu->Project->Settings->C/C++->Preprocessor
// "$(QLIB_INC)" to the additional include directories entry.
// (3) Add in the Dialog Menu->Project->Settings->Linker->General
// "$(QLIB_LIB)\qlib32.lib" to the additional library and object
// modules directories entry.

#include "stdafx.h"
#include "windows.h"
#include "qlib.h"
#include <conio.h>
#include <stdio.h>

#define OUT1     0x1
#define OUT2     0x2
#define OUT3     0x4
#define OUT4     0x8
#define OUT5     0x10
#define OUT6     0x20
#define OUT7     0x40
#define OUT8     0x80
#define OUT9     0x100
#define OUT10    0x200
#define OUT11    0x400
#define OUT12    0x800
#define OUT13    0x1000
#define OUT14    0x2000
#define OUT15    0x4000
#define OUT16    0x8000




int main(int argc, char* argv[])
{

        ULONG handle;
        char ch;

        //
        // Open the USB Module (USBOPTOREL16 or USBOPTO16IO )
        //
        handle = QAPIExtOpenCard(USBOPTOREL16,0);
        if ( handle == 0 ) {
                handle = QAPIExtOpenCard(USBOPTO16IO,0);
        }

        //
        // If there are no usb modules terminate application
        //
        if ( handle == 0 ) {
                MessageBox(NULL, "No USB Modules found!","Error", MB_OK);
                return FALSE;
        }

        // Ok, we found a QUANCOM USB Module

        // ------------------------------------------------------------------
```

© QUANCOM Informationssysteme GmbH

```
        // PART 1: Setting the outputs
        //
        // The following constants can be used to program the outputs:
        // ----------------------------------------------------------------------
        ULONG lines = 0;

        //
        // Reset all lines to "Low"
        //
        printf("Reset all lines to 'Low' ( Press return to continue ):\n");

        QAPIExtWriteDO16(handle, 0, lines, 0);
        ch = getchar();

        //
        // Set the outputs OUT4,OUT10 and OUT15 to "High" ( 16-Bit )
        //
        printf("Set OUT4,OUT10 and OUT15 to 'High' ( Press return to continue ):\n");

        lines = OUT4 + OUT10 + OUT15;
        QAPIExtWriteDO16(handle, 0, lines, 0);
        ch = getchar();

        //
        // Set the output OUT1, OUT4,OUT10 and OUT15 to "High" ( 16-Bit )
        //
        printf("Set OUT1, OUT4,OUT10 and OUT15 to 'High' ( Press return to continue ):\n");

        lines = OUT1 + OUT4 + OUT10 + OUT15;
        QAPIExtWriteDO16(handle, 0, lines, 0);
        ch = getchar();




        //
        // Reset line OUT10 to "Low"
        //
        printf("Reset line OUT10 to 'Low' ( Press return to continue ):\n");
        QAPIExtWriteDO1(handle, 10 - 1, FALSE, 0);
        ch = getchar();

        //
        // Set line OUT5 to "High"
        //
        printf("Set line OUT5 to 'High' ( Press return to continue ):\n");
        QAPIExtWriteDO1(handle, 5 - 1, TRUE, 0);
        ch = getchar();

        //
        // Reset all lines to "Low"
        //
        printf("Reset all to 'Low' ( Press return to continue ):\n");
        lines = 0;
        QAPIExtWriteDO16(handle, 0, lines, 0);
        ch = getchar();


        // ----------------------------------------------------------------------
        // PART 2: Reading the inputs ( and detect changed inputs )
        //
        // This part reads the state of the input lines.
        //
        // The following constants can be used to program the inputs:
        // ----------------------------------------------------------------------
        int i;
        while (!kbhit()) {
                        // read the current state from the inputs
                        lines = QAPIExtReadDI16(handle, 0, 0);
                        printf("\n--------------------------------------------------------------------------------\n");
                        printf("Current input states\n");
                        printf("IN1  IN2  IN3  IN4  IN5  IN6  IN7  IN8  IN9  IN10 IN11 IN12 IN13 IN14 IN15\n");
                        for (i=0;i<15;i++) {
                                if ( lines & 1<<i) {
                                        printf(" 1   ");
                                } else {
                                        printf(" 0   ");
                                }
                        }

                        // read the state change ff from the input lines
                        lines = QAPIExtSpecial(handle, JOB_READ_IN_FFS, 0, 0);
                        printf("\n--------------------------------------------------------------------------------");
                        printf("\nInput change detection 1 = Input has changed its state since last reading\n");
                        printf("IN1  IN2  IN3  IN4  IN5  IN6  IN7  IN8  IN9  IN10 IN11 IN12 IN13 IN14 IN15\n");
                        for (i=0;i<15;i++) {
                                if ( lines & 1<<i) {
                                        printf(" 1   ");
                                } else  {
                                        printf(" 0   ");
```

```
                        }
                }

                printf("\n--------------------------------------------------------------------------------\n");
                printf("Press return to stop reading the inputs every 5 seconds ...\n");
                for (int j=0;(j<10) && !kbhit();j++) {
                        Sleep(500);
                }
        }
        ch = getchar();

        // ------------------------------------------------------------------
        // PART 3: Enabling and disabling the timeout control
        //
        // The timeout control is a security feature. If the application
        // crashes or fails to set the outputs in a predefined time
        // the module resets the outputs to 'low'. This is to prevent that
        // connected devices ( i.e. motors, heaters, lamps, ... )
        // continue to run when the software has lost control.
        // ------------------------------------------------------------------

        printf("Testing timeout function ...\n");
        printf("Set OUT1,OUT2,OUT5,OUT10 and OUT15 to 'high'\n");

        // writing to OUTx pins
        QAPIExtWriteDO16(handle, 0, OUT1+OUT2+OUT5+OUT10+OUT15, 0);
        printf("Setting timout time to 22.4s\n");

        // first disable timeout
        QAPIExtSpecial(handle, JOB_DISABLE_TIMEOUT, 0, 0);

        // now, set timeout time ( 6 = 22.4s )
        lines = QAPIExtSpecial(handle, JOB_SET_WATCHDOG_TIME, 6, 0);
        printf("Enabling the timout control.\n");

        // reset timeout status bit, this may be set by previous testing
        QAPIExtSpecial(handle, JOB_RESET_TIMEOUT_STATUS, 0, 0);

        // enable timeout control
        QAPIExtSpecial(handle, JOB_ENABLE_TIMEOUT, 0, 0);

        // this part waits for 40s without any write to the usb module
        // after 22.4 seconds the hardware watchdog on the board
        // sets all outputs to 'low'
        printf("Now we wait for 40 seconds. This simulates a crash !\n");
        for (int j=0;j<40;j++) {
                Sleep(1000);
                printf(".");
        }

        printf("\nAll OUT Pins should be reset now ! Press a key to continue !");
        ch = getchar();

        ULONG status = QAPIExtSpecial(handle, JOB_READ_TIMEOUT_STATUS, 0, 0);
        if ( status )
                printf("Timeout flag was set\n");
        else
                printf("No timeout flag set\n");



        // first disable timeout
        QAPIExtSpecial(handle, JOB_DISABLE_TIMEOUT, 0, 0);

        // reset timeout status bit
        QAPIExtSpecial(handle, JOB_RESET_TIMEOUT_STATUS, 0, 0);
        printf("\nReady ! Press a key to continue !");
        ch = getchar();

        QAPIExtCloseCard(handle);

        return 0;
}
```

© QUANCOM Informationssysteme GmbH

6.2.1.2 Programming the photocoupled/relays outputs with the QLIB under C

```
#include <windows.h>
#include <stdio.h>
#include <conio.h>

#include "qlib.h"

/*=====================
Main program
=====================*/

void main () {
        ULONG handle;    /*Handle of the USBOPTOREL16

        if ((handle=QAPIExtOpenCard(USBOPTOREL16,0L)) == 0L) {
                printf("Couldn´t open USBOPTOREL16 \n");
                return;
        }

        for (;;) {
                if (kbhit() != 0 && getch() == 27) break;

                QAPIExtWriteDO8(handle,0L,0x00L,0L);
                Sleep(500);
                QAPIExtWriteDO8(handle,0L,0xFFL,0L);
                Sleep(500);
                QAPIExtWriteDO8(handle,0L,0x55L,0L);
                Sleep(500);
                QAPIExtWriteDO8(handle,0L,0xAAL,0L);
                Sleep(500);
        }

        QAPIExtCloseCard(handle);

}
```

6.2.1.3 Read the photo-coupled inputs with the QLIB under C

```
#include <windows.h>
#include <stdio.h>
#include <conio.h>

#include ”qlib.h"


/*=====================
Main program
=====================*/

void main () {
        ULONG handle;    /* Handle of the USBOPTOREL16 */

        if ((handle = QAPIExtOpenCard(USBOPTOREL16,0L)) == 0L) {
                printf("Couldn’t open USBOPTOREL16 \n");
                return;
        }

        for (;;) {
                if (kbhit() != 0 && getch() == 27) break;

                printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
                Sleep(500);
                printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
                Sleep(500);
                printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
                Sleep(500);
                printf("%04lx\n",QAPIExtReadDI16(handle,0L,0L));
                Sleep(500);
        }

        QAPIExtCloseCard(handle);
}
```

© QUANCOM Informationssysteme GmbH

6.2.2 Program examples in Borland Delphi for the QLIB

After the installation of the QLIB, you'll find the example source (usbmodul.pas) at the folder d:\program files\quancom\qlib32\samples\usbopto\delphi (only if you had installed to the standard folder).

© QUANCOM Informationssysteme GmbH

6.2.2.1 Programming the Inputs, Outputs and special functions

```
program USBOPTO_CONSOLE_APP;

uses
   WinProcs;
{*
// usbopto.pas : Sample project for the USBOPTOREL16 and USBOPTOIO Modules
//
// Author: Michael Reimer, QUANCOM Informationssysteme GmbH, Germany
//
// Website: http://www.quancom.de
// Product:
// USB OPTO I/O Module http://www.quancom.de/qprod01/eng/pb/usbopto16io.htm
// USB OPTO Relay Module http://www.quancom.de/qprod01/deu/pb/usboptorel16.htm
// Information:
//
// To use the QLIB Commands in your source, do the following:
//
// (1) Add statement #include "qlib.pas" to your source file.
// (2) Copy QLIB.PAS from QLIB Installation Directory
//     d:\program files\quancom\qlib32\include to your
//     working directory
*}
function KeyPressed: Boolean;
var
 Buffer: TInputRecord;
 Data: DWORD;
 hIn: DWORD;

begin
 hIn := GetStdHandle( STD_INPUT_HANDLE );
 Result := False;
 PeekConsoleInput(hIn, Buffer, 1, Data);
 if Data <> 0 then
   if Buffer.EventType = Key_Event then
       begin
            Result := True;
       end;

 FlushConsoleInputBuffer(hIn)

end;

{$APPTYPE CONSOLE}

{$INCLUDE QLIB.pas}

{$X+}

const OUT1 = $1;
      OUT2 = $2;
      OUT3 = $4;
      OUT4 = $8;
      OUT5 = $10;
      OUT6 = $20;
      OUT7 = $40;
      OUT8 = $80;
      OUT9 = $100;
      OUT10 = $200;
      OUT11 = $400;
      OUT12 = $800;
      OUT13 = $1000;
      OUT14 = $2000;
      OUT15 = $4000;
      OUT16 = $8000;

var handle: longint;
    status: longint;
    lines: longint;
    s: string;
    i: integer;
    j: integer;

Label lab1, lab2;

begin

{*
//
// Open the USB Module
//
*}

handle := QAPIExtOpenCard(USBOPTOREL16,0);
if ( handle = 0 ) then
   begin
   handle := QAPIExtOpenCard(USBOPTO16IO,0);
```

```
   end;

{*
// The handle is <> NULL if there is a USB Module installed
*}

if ( handle = 0 ) then
   begin
       s := 'No QUANCOM USB Module found!';
       writeln(s);
       halt(0);
   end;

{*
// Ok, we found a QUANCOM USB Module



// ---------------------------------------------------------------------------
// PART 1: Setting the outputs
//
// The following constants can be used to program the outputs:
// ---------------------------------------------------------------------------
*}

{*
//
// Reset all lines to "Low"
//
*}

lines := 0;

writeln('Reset all lines to Low ( Press return to continue ):');

QAPIExtWriteDO16(handle, 0, lines, 0);

Readln;

{*
//
// Set the outputs OUT4,OUT10 and OUT15 to "High" ( 16-Bit )
//
*}

writeln('Set OUT4,OUT10 and OUT15 to High ( Press return to continue ):');

lines := OUT4 + OUT10 + OUT15;

QAPIExtWriteDO16(handle, 0, lines, 0);

Readln;

{*
//
// Set the output OUT1, OUT4,OUT10 and OUT15 to "High" ( 16-Bit )
//
*}

writeln('Set OUT1, OUT4,OUT10 and OUT15 to High ( Press return to continue ):');

lines := OUT1 + OUT4 + OUT10 + OUT15;

QAPIExtWriteDO16(handle, 0, lines, 0);

Readln;

{*
//
// Reset line OUT10 to "Low"
//
*}

writeln('Reset line OUT10 to Low ( Press return to continue ):');

QAPIExtWriteDO1(handle, 10 - 1, 0, 0);

Readln;

{*
//
// Set line OUT5 to "High"
//
*}

writeln('Set line OUT5 to High ( Press return to continue ):');

QAPIExtWriteDO1(handle, 5 - 1, 1, 0);

Readln;
```

© QUANCOM Informationssysteme GmbH

```
{*
//
// Reset all lines to "Low"
//
*}

writeln('Reset all to Low ( Press return to continue ):');

lines := 0;

QAPIExtWriteDO16(handle, 0, lines, 0);

Readln;

{*
// ---------------------------------------------------------------------
// PART 2: Reading the inputs ( and detect changed inputs )
//
// This part reads the state of the input lines.
//
// The following constants can be used to program the inputs:
// ---------------------------------------------------------------------
*}

while Not KeyPressed do
      begin
      {*
      // read the current state from the inputs
      *}

      lines := QAPIExtReadDI16(handle, 0, 0);

      write(#13#10'-------------------------------------------------------------------------------'#13#10);
      write('Current input states'#13#10);
      write('IN1  IN2  IN3  IN4  IN5  IN6  IN7  IN8  IN9  IN10 IN11 IN12 IN13 IN14 IN15 IN16'#13#10);

      for i:=0 to 15 do
          begin
          if lines AND ( 1 shl i)  <> 0 then
              begin
                   write(' 1   ');
                   end
          else
              begin
                   write(' 0   ');
                   end
          end;

      {*
      // read the state change ff from the input lines
      *}

      lines := QAPIExtSpecial(handle, JOB_READ_IN_FFS, 0, 0);

      write(#13#10'-------------------------------------------------------------------------------');
      write(#13#10'Input change detection 1 = Input has changed its state since last reading'#13#10);
      write('IN1  IN2  IN3  IN4  IN5  IN6  IN7  IN8  IN9  IN10 IN11 IN12 IN13 IN14 IN15 IN16'#13#10);

      for i:=0 to 15 do
          begin
          if lines AND ( 1 shl i)  <> 0 then
              begin
                   write(' 1   ');
                   end
          else
              begin
                   write(' 0   ');
                   end
          end;

      write(#13#10'-------------------------------------------------------------------------------'#13#10);
      write('Press return to stop reading the inputs every 5 seconds ...'#13#10);

      for j:=0 to 10 do
          begin
           Sleep(500);
           If KeyPressed then
              goto lab1;
          end;
      end;

lab1:

{*
// ---------------------------------------------------------------------
// PART 3: Enabling and disabling the timeout control
//
// The timeout control is a security feature. If the application
// crashes or fails to set the outputs in a predefined time
```

```
// the module resets the outputs to 'low'. This is to prevent that
// connected devices ( i.e. motors, heaters, lamps, ... )
// continue to run when the software has lost control.
// ----------------------------------------------------------------------
*}

writeln('Testing timeout function ...'#13#10);

writeln('Set OUT1,OUT2,OUT5,OUT10 and OUT15 to High'#13#10);

{*
// writing to OUTx pins
*}

QAPIExtWriteDO16(handle, 0, OUT1+OUT2+OUT5+OUT10+OUT15, 0);

writeln('Setting timout time to 22.4s'#13#10);

{*
// first disable timeout
*}

QAPIExtSpecial(handle, JOB_DISABLE_TIMEOUT, 0, 0);

{*
// now, set timeout time ( 6 = 22.4s )
*}

QAPIExtSpecial(handle, JOB_SET_WATCHDOG_TIME, 6, 0);

writeln('Enabling the timout control.'#13#10);

{*
// reset timeout status bit, this may be set by previous testing
*}

QAPIExtSpecial(handle, JOB_RESET_TIMEOUT_STATUS, 0, 0);

{*
// enable timeout control
*}

QAPIExtSpecial(handle, JOB_ENABLE_TIMEOUT, 0, 0);

{*
// this part waits for 40s without any write to the usb module
// after 22.4 seconds the hardware watchdog on the board
// sets all outputs to 'low'
*}

writeln('Now we wait for 40 seconds. This simulates a crash !'#13#10);

for j := 0 to 40 do
    begin
         Sleep(1000);
         write('.');
         If KeyPressed then
            goto lab2
    end;

lab2:

writeln(#13#10'All OUT Pins should be reset now ! Press a key to continue !');

ReadLn;

status := QAPIExtSpecial(handle, JOB_READ_TIMEOUT_STATUS, 0, 0);

if  status <> 0 then
    begin
      writeln('Timeout flag was set'#13#10);
      end
else
    begin
      writeln('No timeout flag set'#13#10);
      end;


{*
// first disable timeout
*}

QAPIExtSpecial(handle, JOB_DISABLE_TIMEOUT, 0, 0);

{*
// reset timeout status bit
*}

QAPIExtSpecial(handle, JOB_RESET_TIMEOUT_STATUS, 0, 0);
```

© QUANCOM Informationssysteme GmbH

```
writeln(#13#10'Ready ! Press a key to continue !');
ReadLn;
QAPIExtCloseCard(handle);

end.
```

© QUANCOM Informationssysteme GmbH

7. Annex

7.1 Problems with boards running under Windows 98/95 and Windows XP/2000/NT

Why is the ”Control Panel” board configuration dialog ”QLIB” empty?

• There is no QUANCOM PCI board in the system.

• There are no drivers installed for a QUANCOM ISA board.

I get the message ”QLIBNDRV.SYS not found ”or ”QLIBNDRV.VXD not found” after installation. What can I do?

• Check that the QLIB is installed properly. For further information about the installation process and the general programming with the QLIB please see the “ QLIB” manual which is included on the installation CD.

• If you use a QUANCOM ISA board check if the drivers for the QUANCOM board are installed.

Why do I get the message "Driver QLIBNDRV.SYS” or ”Driver QLIBNDRV.VXD” could not be load?

• Check that the QLIB is installed properly. For further information about the installation process and the general programming with the QLIB please read the “ QLIB” manual which is included on the installation CD.

• The driver for the QUANCOM board was not loaded. (Control Panel => System)

Windows XP/2000/NT: Does the QLIB have to be installed with Administrator-right?

• Yes, always install the QLIB with administration rights.

Windows XP/2000/NT: Why do I get the message ”Driver could not be installed” during the installation?

• Installation was made without administration rights.

© QUANCOM Informationssysteme GmbH

Windows XP/2000/NT: Why do I get the message "Driver QLIBNDRV.SYS could not be loaded”?

• The driver’s installation has failed, because the QLIB was not installed with administration-rights.

• QLIB-Software was installed on a network drive. Always install the QLIB on your local drive.

Windows XP/2000/NT: How can I manually install the driver QLIBNDRV.SYS?

If the QLIBNDRV.SYS failed to install, it may be necessary to install the driver manually.

Please take the following steps to install the driver manually:

• Search on the installation CD ”Tools” for the tool instdrv.exe in the directory. With this tool you can install and de-install the driver manually.

• Please call this tool with the following command line parameters:

instdrv qlibndrv d:\directory\qlibndrv.sys

• Go to ”Start -> Settings ->Control panel ->(Administrative Tools / Windows 2000 only) -> Drivers” change the start type to ”Automatic”, then click on the ”Start” button. Please restart the system for the changes to become active.

Why do I have to restart the driver after every reboot?

The starting type of the driver is set to ”Manual”. If you wanted to you are able change this setting on “Automatic” to start the driver on every reboot of the system.

© QUANCOM Informationssysteme GmbH

7.2 Customer Communication and Help

Did you need help?

If you don't know how to go on during the installation or operation of your QUANCOM board please consult this user's guide first.

! Tip !

You can find an ASCII – text – file README.TXT, which includes changes made after printing of this user's manual on the QUANCOM installation CD.

! Important !

Informationen bereit: If you have further questions please contact our support team. For this case please prepare the following information:

- Exact type of the board.
- Version of the driver
- Version of the QLIB
- Operating system, hardware equipment and bus - system
- Name and version of the program, which reports the failure
- A detailed failure description. To make sure, please try to reproduce the failure, and describe exactly, which steps led to this failure.

Contact?

Die QUANCOM Internet Webseite
www.quancom.deThe QUANCOM internet website
http://www.quancom.de/

© QUANCOM Informationssysteme GmbH

Per Fax
+49 22 36 / 89 92 - 49

Per E-Mail:
support@quancom.de

Adress:
QUANCOM INFORMATIONSSYSTEME GmbH
In der Flecht 14
50389 Wesseling

Wenn Sie Hilfe brauchen, erreichen Sie uns unter:
QUANCOM Hotline Deutschland If you need urgent help call:QUANCOM Hotline Germany
0 22 36 / 89 92 - 20

Monday-Thursday
from 9:00 to 18:00
Friday
from 9:00 to 17:00

© QUANCOM Informationssysteme GmbH

Aktuelle Treiber

You can find the latest version of QUANCOM software on our internet website http://www.quancom.de. You can also find a lot of information and "Frequently asked questions (FAQ's)" there., please check if you are using the latest software version of the QUANCOM software before contacting the QUANCOM support.

Reparatur

If you are not sure whether your QUANCOM board is defective please call the QUANCOM Hotline:

Tel.: **+49 22 36 / 89 92 – 20**

Before sending us the QUANCOM board to be repaired call:

Tel.: **+49 22 36 / 89 92 – 20**

If you send your QUANCOM board to us, please use original package or any other suitable package, to protect the contents against transport damage. You also need to send us a copy of the original bill and the RMA number.

You can shorten the repair time by sending us an exact failure description, so that a faster failure search is possible. Send your QUANCOM board directly to the service department of QUANCOM Informations-systeme GmbH.

7.3 Technical support form

If you have internet access please enter the following URL in your browser: **http://www.quancom.de/quancom/qshop.nsf/techniksupport?OpenForm&eng**
else photocopy this form and use the copy of this form as a reference for your current configuration. Complete this form before contacting QUANCOM Informationssysteme GmbH for technical support help and our applications engineers may answer your questions more

© QUANCOM Informationssysteme GmbH

efficiently. If you are using any other QUANCOM hardware or software products please add them to this configuration form. Include additional pages if necessary.

Name: _____

Company: _____

Address: _____

Phone: _____

Fax: _____

Computer brand / Processor: _____

Operating system: _____

Display adapter: _____

Mouse: _____

QUANCOM board: _____

Other adapters installed: _____

Hard disk (capacity, free): _____

The problem is: _____

List any error messages: _____

_____

_____

The following steps will reproduce the problem

_____

_____

_____

_____

_____

7.4 Hardware and software configuration form

This form allows you to record the settings of your hardware and software. Complete this form each time you revise your software or hardware configuration, and use this form as a reference for your current configuration. Completing this form accurately before contacting QUANCOM Informationssysteme GmbH for technical support helps our application engineers answer your questions more efficiently.

© QUANCOM Informationssysteme GmbH

- **QUANCOM Product**

Name / Name of board: _____

Interrupt Level: _____

DMA Channel: _____

Base I/O Address: _____

Operating system: _____

- **Other information**

Computer brand and Model: _____

Processor: _____

Clock Frequency: _____

Type of Video Board Installed: _____

DOS Version: _____

Programming Language: _____

Programming Language Version: _____

- **Other Boards in System**

Base I/O Address of other Boards: _____

DMA Channels of other Boards: _____

Interrupt Level of other Boards: _____

7.5 Documentation comment form

QUANCOM Informationsysteme GmbH would like you to comment on the documentation supplied with our products. This information helps us to provide you with quality products to meet your needs. Include additional pages if necessary.

Titel: USBOPTO/REL 64/32/16/8

Edition date: 15.04.2009

© QUANCOM Informationssysteme GmbH

Name: _____

Firma: _____

Adresse: _____

Telefon: _____

Fax: _____

Kommentar: _____

_____

_____

_____

_____

Email to: **support@quancom.de**

Fax to: **+49 2236 89 92 49**

Adress: **QUANCOM Informationssysteme GmbH**
In der Flecht 14
50389 Wesseling

7.6 Trademarks

Linux is registered trade-mark of Linus Torvalds.

MS, MS-DOS, Microsoft, Visual Basic, Windows, Windows Vista/XP/2000/NT/ME/98/95 is registered trade-mark of Microsoft Corporation.

XT and PS/2 are trade-marks and IBM, OS/2 and AT are registered trade-mark of International Business Machines Corporation.

Intel, Pentium is registered trade-mark of Intel Corporation.

USB is registered trade-mark of USB Implementers Forum Inc.

JAVA is registered trade-mark of Sun Microsystems.

DELPHI and Pascal are registered trade-mark of Borland Corporation.

PCI is registered trade-mark of PCI Special Interest Group.

PCI Express is registered trade-mark of PCI-SIG.

National Instruments, LABVIEW is registered trade-mark of National Instruments Corporation.

Agilent VEE is registered trade-mark of Agilent Technologies.

By other product- and company names, that are mentioned in this manual, it may deal with trademarks of the respective owners.

© QUANCOM Informationssysteme GmbH